prostory jmen (no) - úvod

- ”oddělení” názvů proměnných/identifikátorů v rámci různých částí programu
- zabránění kolizí stejných jmen (u různých programátorů) - jsou-li proměnné se stejným názvem v různých jmenných prostorech, potom jejich názvy nekolidují
- použitím jmenného prostoru se přidává ”příjmení” ke jménu (jméno prostoru se přidá k názvu proměnné/funkce jako příjmení a tedy celek příjmení::jméno je různý i pro stejná jména)

prostor::identifikátor
prostor::podprostor::identifikátor

- například proměnné v nové knihovně cstdio leží v prostoru std a tedy při tisku je musíme psát

```
std::printf(“text“); // jsme-li mimo prostor std
```

- pokud jsme ve stejném prostoru jako volaná funkce (nebo využívaná proměnná), nemusíme jméno prostoru používat

```
printf(“text“); // jsme-li v prostoru std
```

prostory jmen (no) – použití volání

- při použití má přednost ”nejbližší” identifikátor – prvně se hledá v aktuálním prostoru, potom v nadřazeném. Do prostorů s jiným jménem se bez uvedení pomocí using nedostaneme
- klíčové slovo using – zpřístupňuje v daném prostoru identifikátory či celé prostory skryté v jiném prostoru. Umožní neuvádět ”příjmení” při přístupu k proměnným z jiného prostoru
- doporučuje se zpřístupnit pouze vybrané funkce a data (ne celý prostor = totální porušení ohraničení=ochrany).
- Zároveň se nedoporučuje používání using (zpřístupňování) v hlavičkových souborech = globální dosah zpřístupnění/otevření (v každém souboru, kde je hlavičkový soubor následně includován). Lépe zpřístupnit v c/cpp souborech jen tam kde je skutečně potřeba
- zpřístupnění/”dotažení” proměnné končí s koncem bloku, ve kterém je uvedeno

nechceme-li psát std:: při volání printf

```
std::printf(“text“);
```

musíme před použitím prostého printf uvést některý z příkazů:

```
using std::printf(char *); // pouze funkce
using namespace std; // celý prostor – horší varianta
```

prostory jmen (no) - vytvoření

- vlastní prostor definujeme pomocí klíčového slova namespace – vyhrazuje (vytváří) prostor s daným jménem
- vytváří blok společných funkcí a dat – oddělených od zbytku (jménem prostoru)
- definičních bloků se stejným názvem prostoru může být v programu více – do uvedeného jmenného prostoru se „přidají“ nové funkce a data (obsah prostoru může být tedy definován na více místech).
- přístup do jiných prostorů přes celý název proměnné

definice:

```
namespace XX { // začátek prostoru s daným jménem XX
proměnné  // definice proměnných patřících do prostoru XX
funkce
třídy
};
```

prostory jmen (no) – použití definice

Hlavičkový soubor

```
namespace XX {
double funkce(void); // funkce v prostoru
struct AA { // struktura definovaná ve jmenném prostoru
double x;
};
}
```

Zdrojový soubor (vložení všeho stejně jako u hlavičky, nebo lze i jednotlivě):

```
double XX::funkce(void) // nutno uvést prostor do kterého patří
{
AA  a; // definice proměnné struktura „uvnitř“ prostoru

 return (a.x);
}

XX::AA b; // definice proměnné struktura mimo prostor
```

prostory jmen (no) - poznámka

- pomocí **using NadřazenýProstor::x; using BázováTřída::g**; lze zpřístupnit prvek, který se ”ztratil” (byl překryt či je přímo nepřístupný)
  Pozn.: „BázováTřída“ – viz. dědění

- pomocí namespace můžeme zkrátit i název vnořených prostorů

```
namespace zkratka = Petr::Prostor1::Oddil2;

// přístup k proměnným v prostoru Oddil z Prostor1 z Petr
x = zkratka::promenna;
```